Programmer avec des objets

Objets et structures de données

- Classe
- Objet avec données
  - Cas simple
  - Liste chaînée
- Objet avec méthodes
- FIFO

Objets et structures de données

- **Classe**
- Objet avec données
  - Cas simple
  - Liste chaînée
- Objet avec méthodes
- FIFO

Ma première classe

```
class truc ():
    pass
```

Ma première classe

```
class truc ():
    pass
```

```
In [2]:  machin = truc()

In [3]:  type(machin)
Out[3]: __main__.truc
```

Objets et structures de données

- Classe
- **Objet avec données**
  - **Cas simple**
  - Liste chaînée
- Objet avec méthodes
- FIFO

Enregistrement

```
class blob ():
    i = 0
    j = -1
    b = True
```

Enregistrement

```
class blob ():
    i = 0
    j = -1
    b = True
```

```
In [2]: chose = blob()

In [3]: chose.i
Out[3]: 0

In [4]: chose.b
Out[4]: True
```

Enregistrement

```
class blob ():
    i = 0
    j = -1
    b = True
```

chose

i	0
j	-1
b	True

Enregistrement

```
class blob ():
    i = 0
    j = -1
    b = True

In [5]: chose.i = 3

In [6]: chose.i
Out[6]: 3
```

chose

i	0
j	-1
b	True

QCM

```
class blob ():
    i = 0
    j = -1
    b = True

A = blob()
B = blob()
A.i = 2
X = A.i + B.i
```

Après l'exécution de ce code, que contient la variable X ?

a) 0

b) 2

c) 3

d) 4

Objets et structures de données

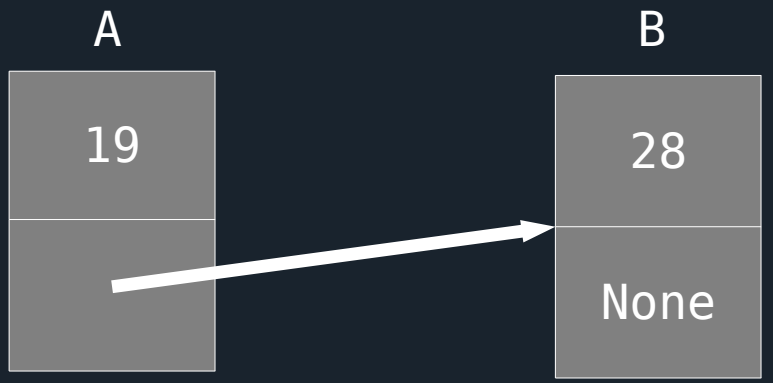
- Classe
- **Objet avec données**
  - Cas simple
  - **Liste chaînée**
- Objet avec méthodes
- FIFO

Liste chaînée

```
class Cellule():
    Valeur = None
    Suivant = None
```

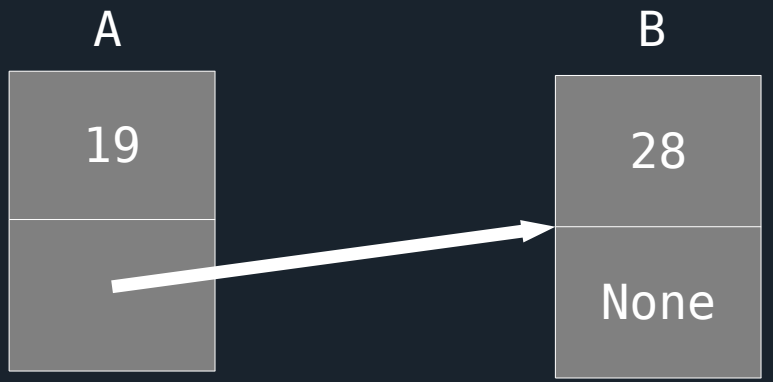
Liste chaînée

```
class Cellule():
    Valeur = None
    Suivant = None
```

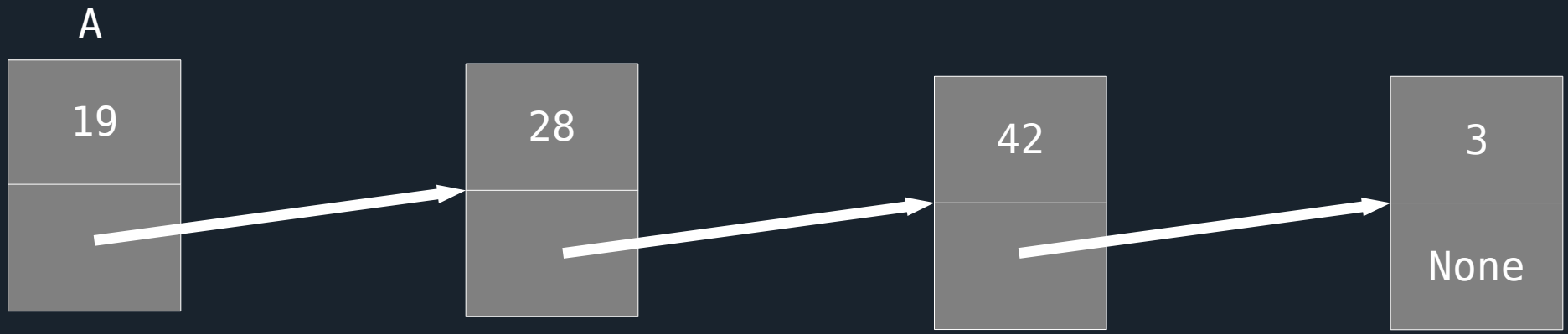


Liste chaînée

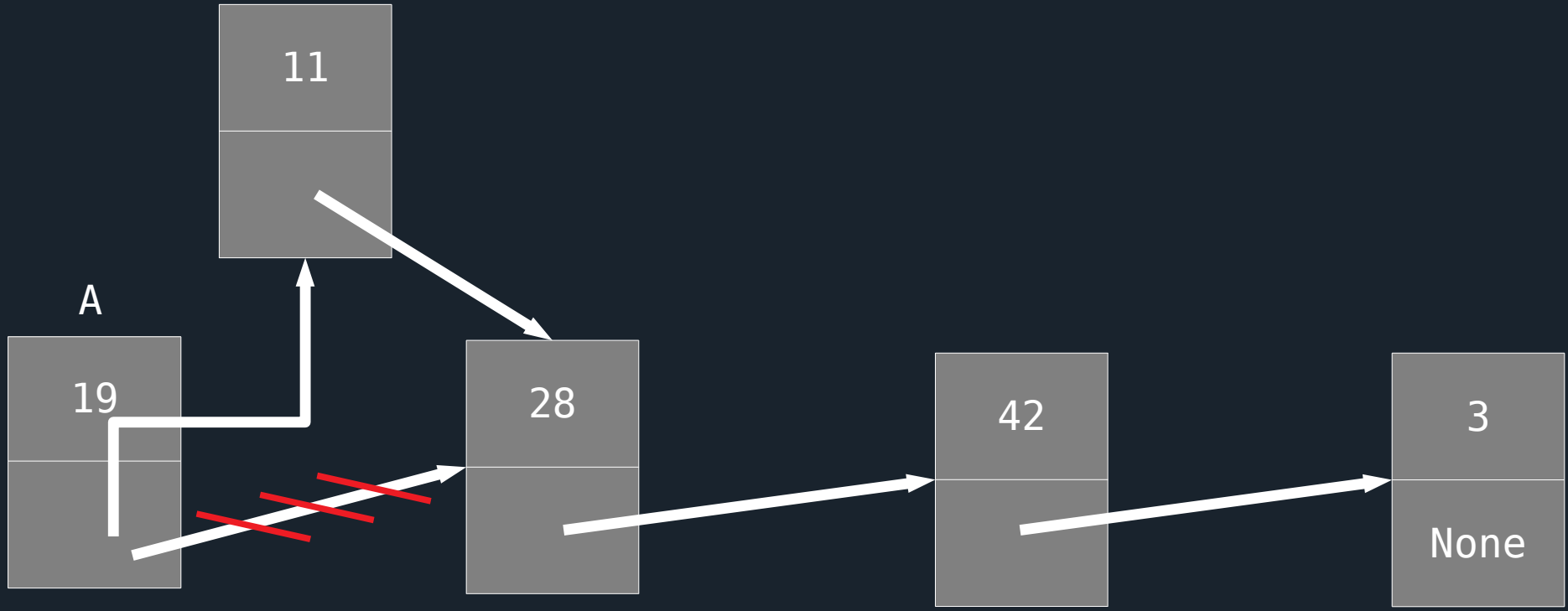
```
class Cellule():
    Valeur = None
    Suivant = None

A = Cellule()
B = Cellule()
A.Valeur = 19
B.Valeur = 28
A.Suivant=B
```



Liste chaînée

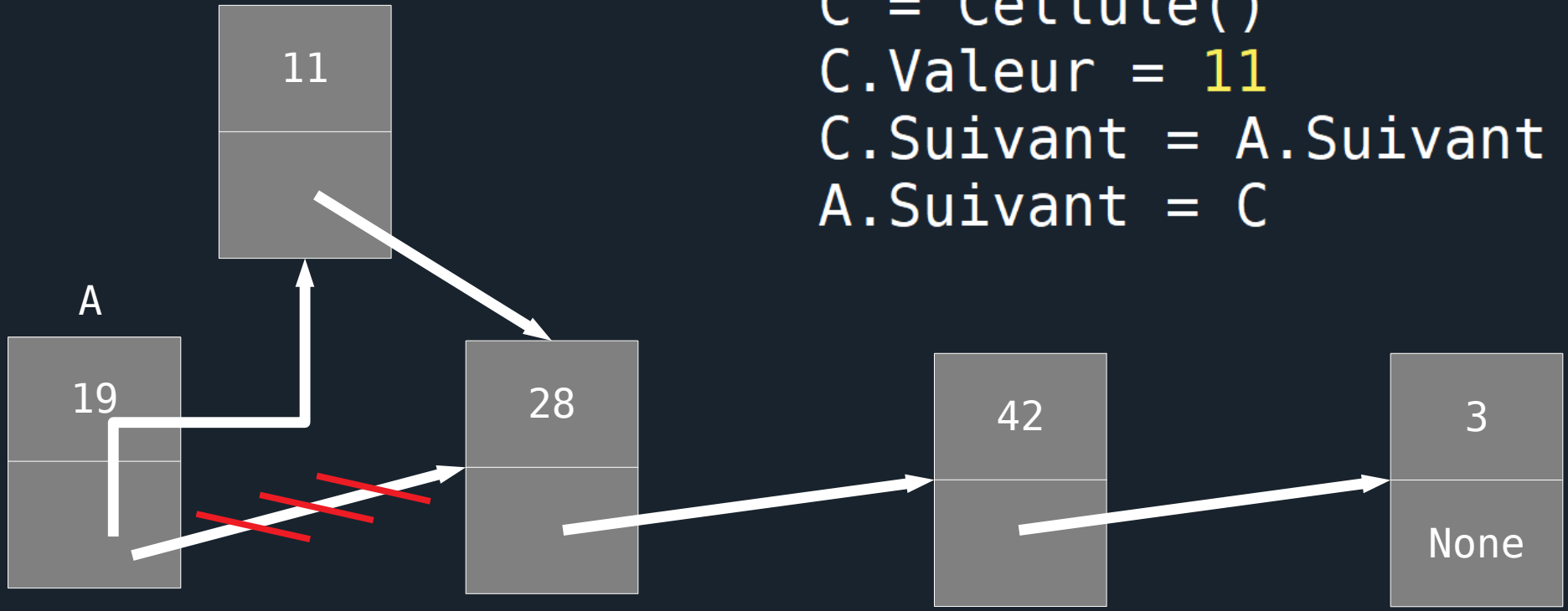


Liste chaînée : insertion

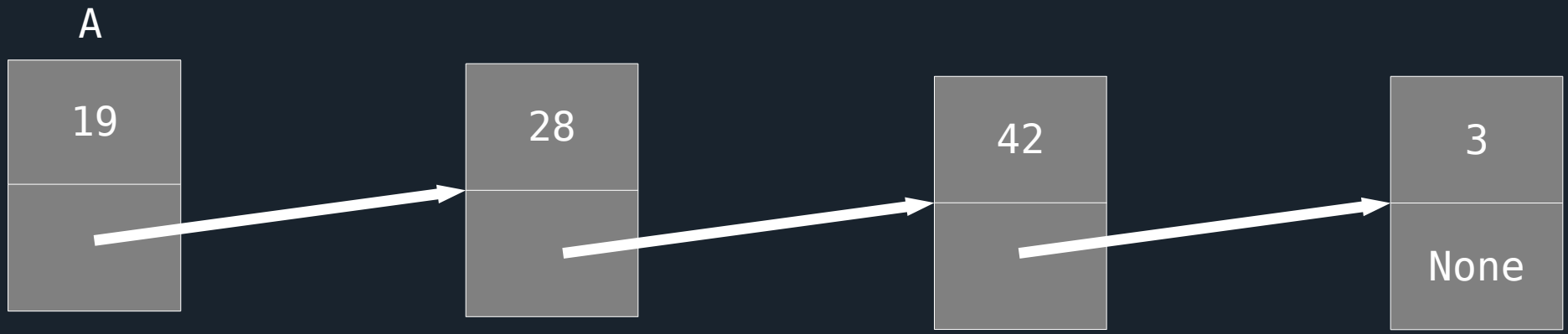


Liste chaînée : insertion

```
C = Cellule()
C.Valeur = 11
C.Suivant = A.Suivant
A.Suivant = C
```

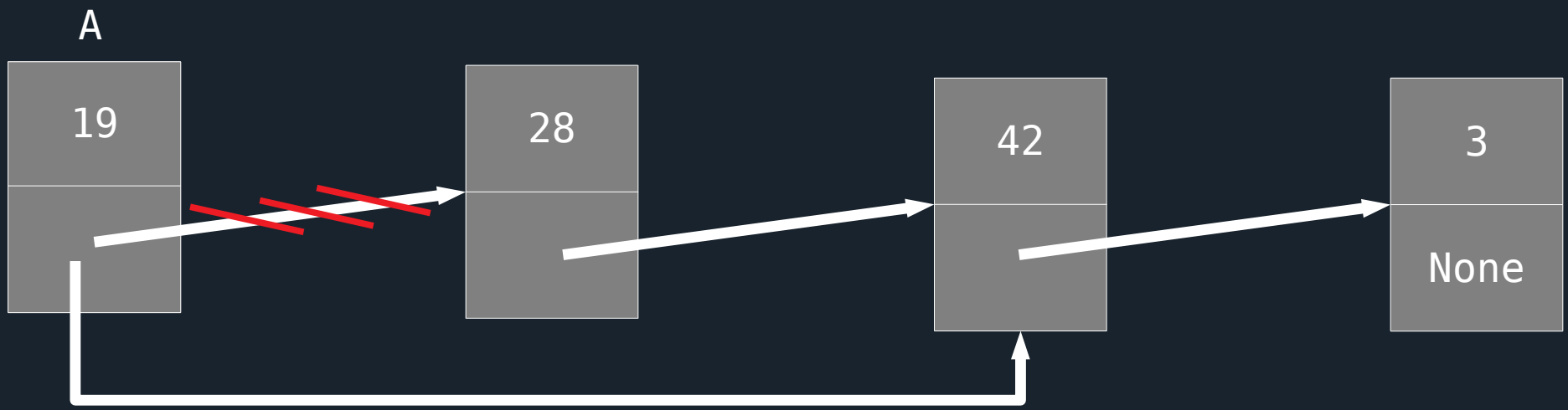


Liste chaînée

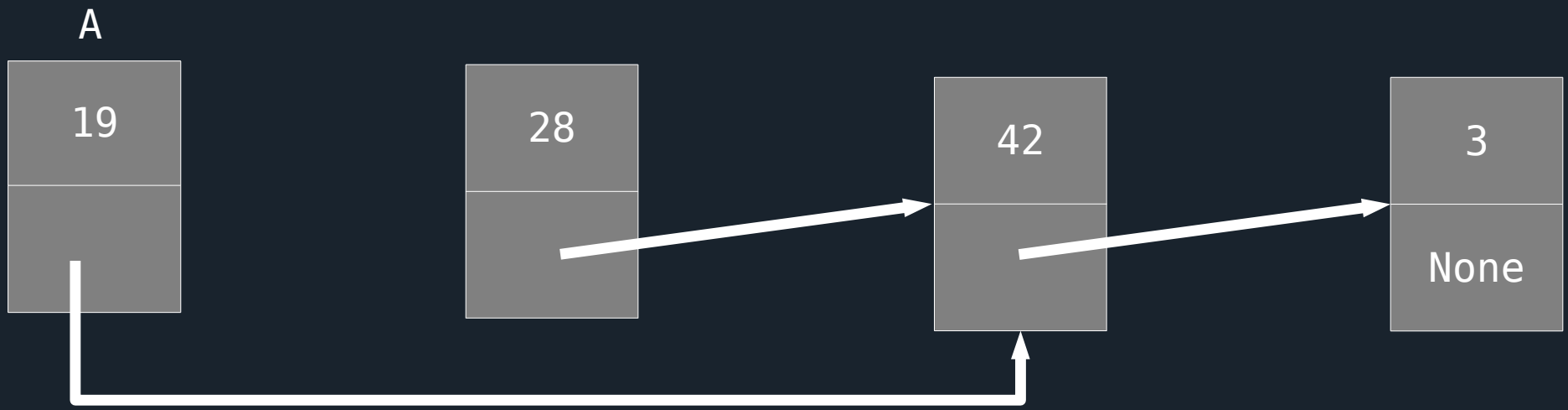


Liste chaînée : suppression

```
A.Suivant = A.Suivant.Suivant
```



Liste chaînée : suppression



QCM

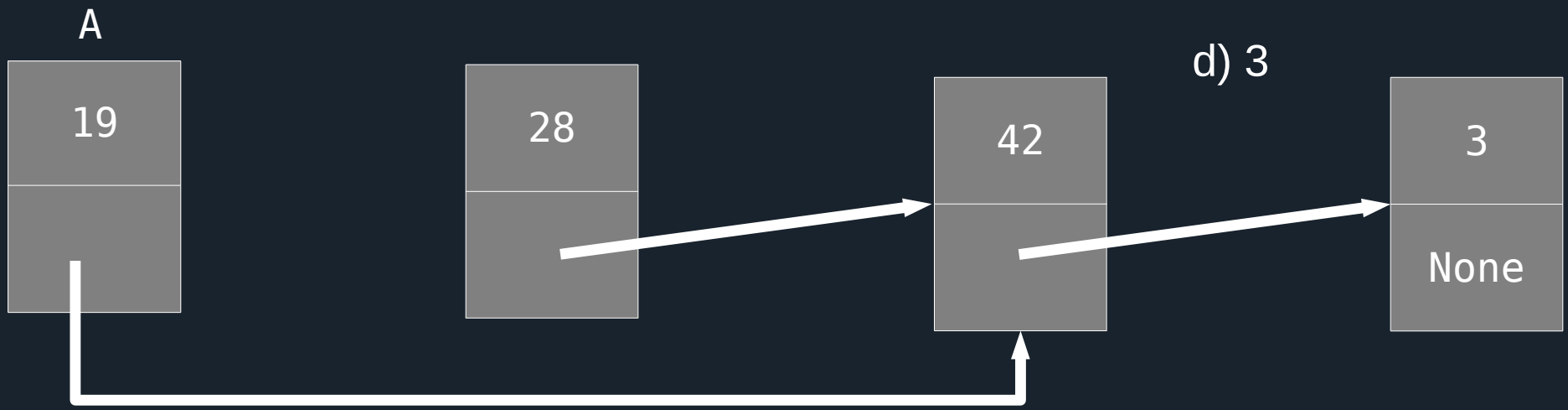
Que vaut A.Suivant.Suivant.Valeur ?

a) 19

b) 28

c) 42

d) 3



Objets et structures de données

- Classe
- Objet avec données
  - Cas simple
  - Liste chaînée
- **Objet avec méthodes**
- FIFO

Compteur

```
class Compteur():
    Valeur = 0

    def incrementer(self):
        self.Valeur = self.Valeur+1
```

Compteur

```
class Compteur():
    Valeur = 0

    def incrementer(self):
        self.Valeur = self.Valeur+1
```

```
In [2]: A = Compteur()

In [3]: A.Valeur
Out[3]: 0

In [4]: A.incrementer()

In [5]: A.Valeur
Out[5]: 1
```

Méthode + argument

```
class Compteur():
    Valeur = 0

    def incrementer(self):
        self.Valeur = self.Valeur+1

    def ajouter(self, n):
        self.Valeur = self.Valeur + n
```

Méthode + argument

```
class Compteur():
    Valeur = 0

    def incrementer(self):
        self.Valeur = self.Valeur+1

    def ajouter(self, n):
        self.Valeur = self.Valeur + n
```

```
In [2]: A = Compteur()

In [3]: A.ajouter(10)

In [4]: A.Valeur
Out[4]: 10
```

Initialisation

```
class Compteur():
    Valeur = 0

    def incrementer(self):
        self.Valeur = self.Valeur+1

    def ajouter(self, n):
        self.Valeur = self.Valeur + n

    def __init__ (self, n):
        self.Valeur = n
```

```
In [2]: A = Compteur(10)

In [3]: A.Valeur
Out[3]: 10
```

QCM

L = [3, 2, 5]

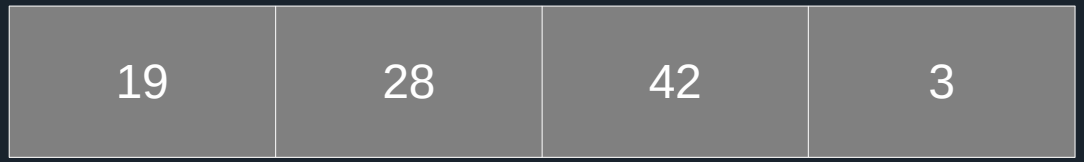
Dans quel cas a-t-on écrit explicitement une méthode sur les listes ?

a) sorted(L)
b) L = L + [7]
c) L.append(7)
d) L[0] = 5

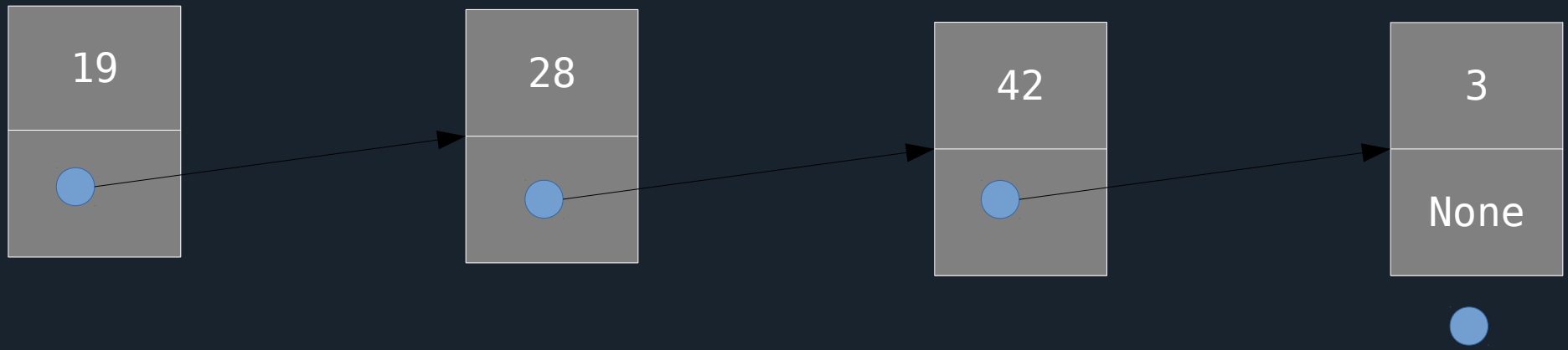
Objets et structures de données

- Classe
- Objet avec données
  - Cas simple
  - Liste chaînée
- Objet avec méthodes
- FIFO

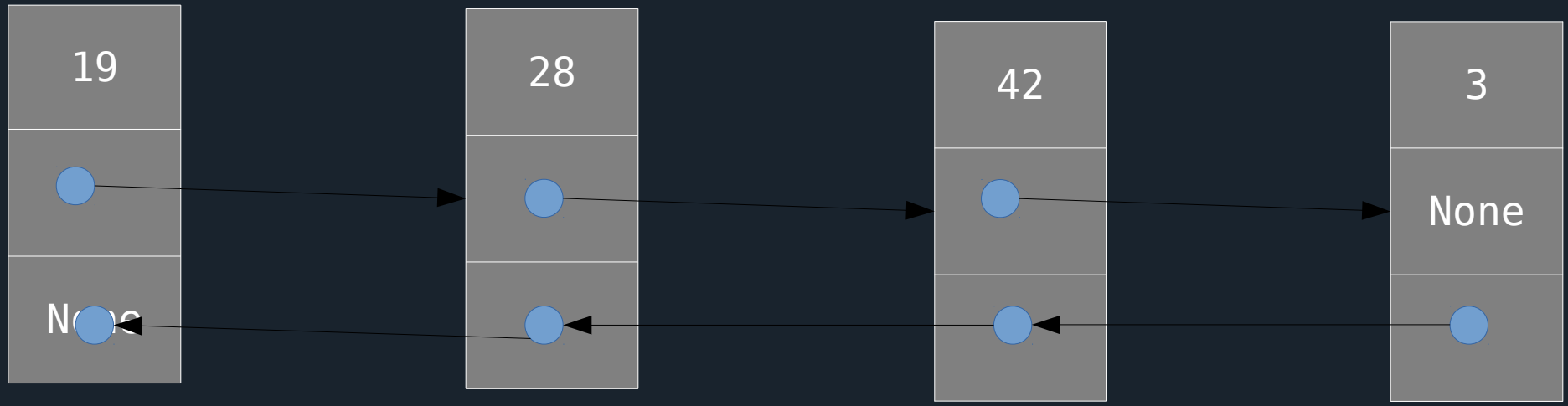
FIFO : First In First Out



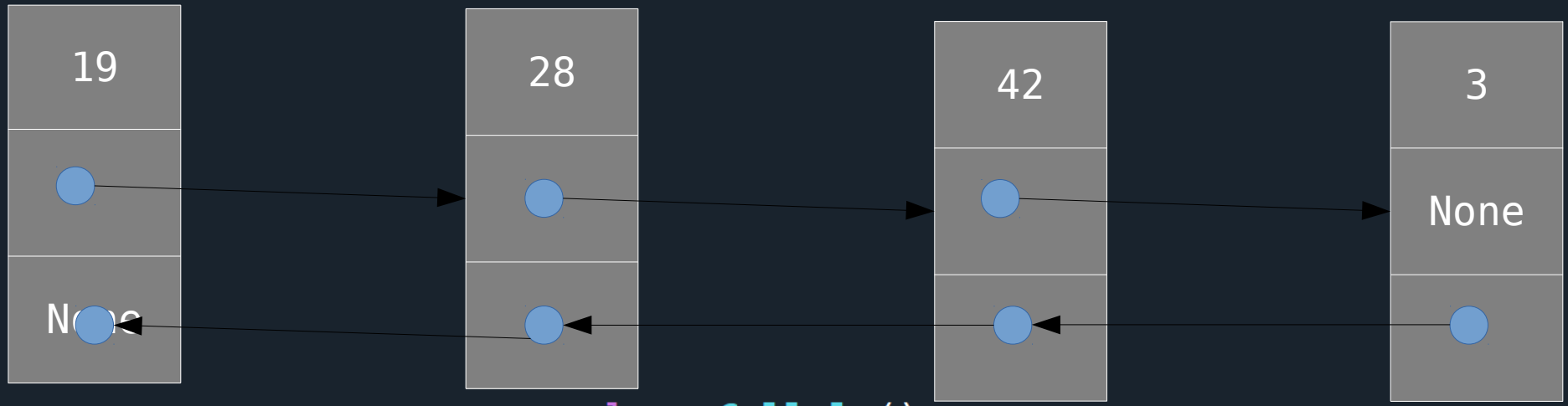
Liste chaînée



Liste doublement chaînée



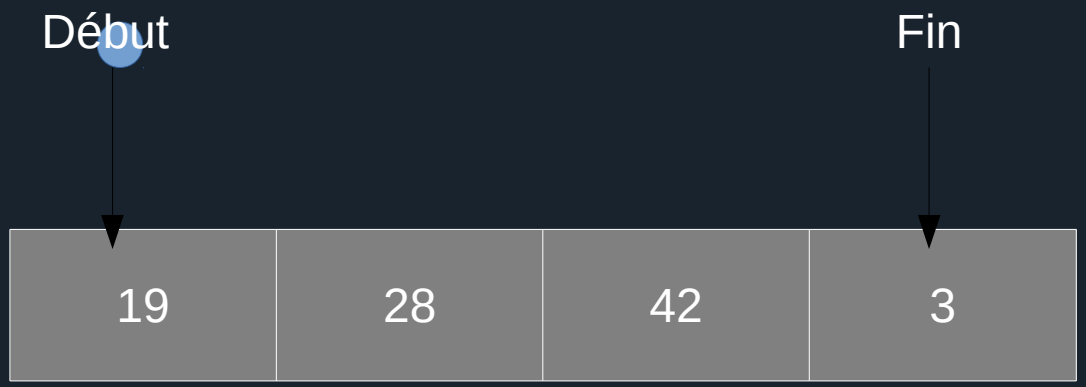
Liste doublement chaînée




```
class Cellule():
    Valeur = None
    Précédent = None
    Suivant = None
```

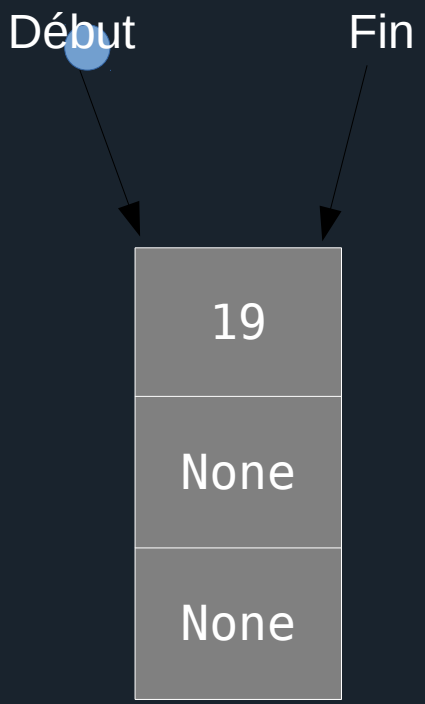
FIFO

```
class FIFO():
    Début = None
    Fin = None
```



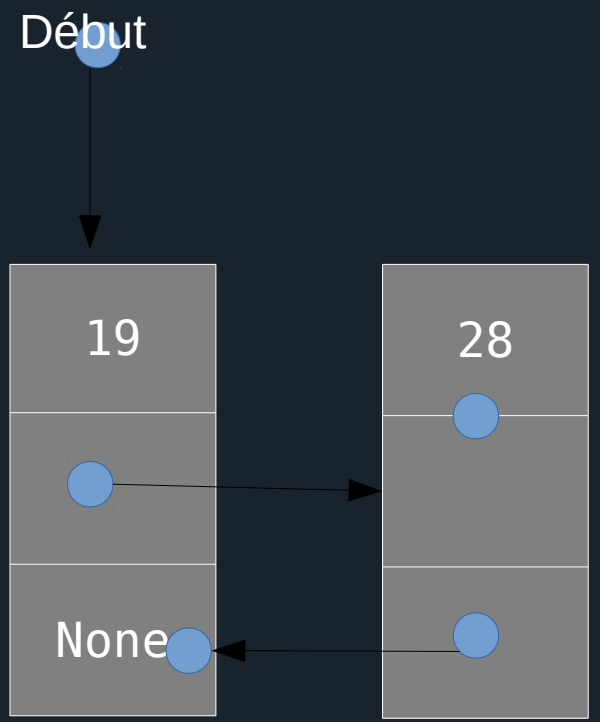
FIFO

```
class Cellule():
    Valeur = None
    Précédent = None
    Suivant = None


class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):
        C = Cellule()
        C.Valeur = v
        self.Début = C
        self.Fin = C
```



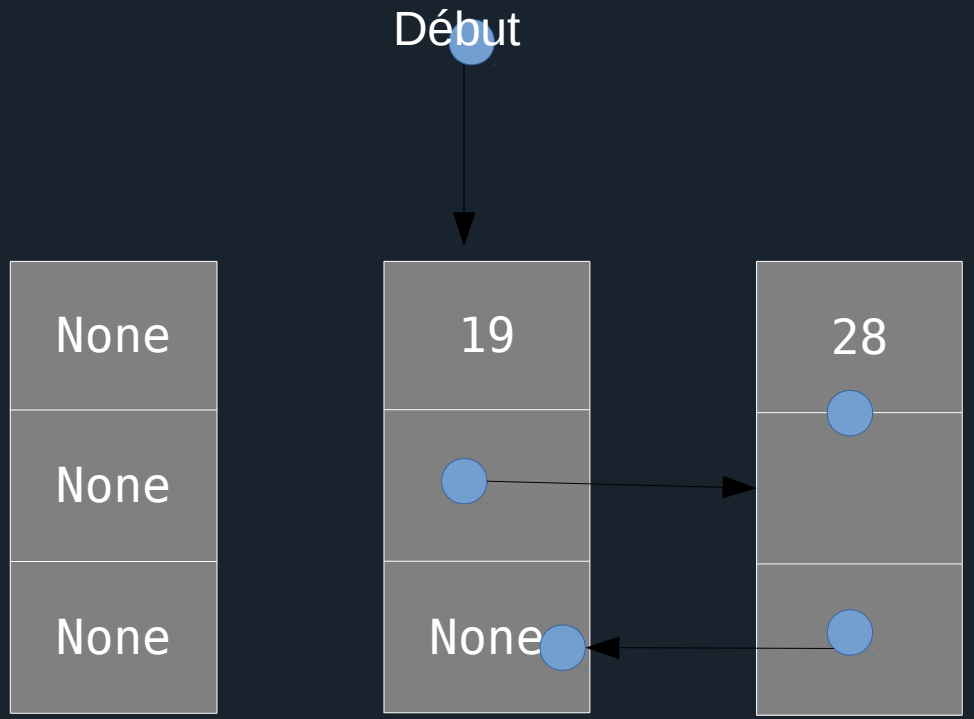
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):
        if self.Début == self.Fin == None:
            self.ajouter_dans_FIFO_vide(v)
        else:
            C = Cellule()
            C.Valeur = v
            C.Suivant = self.Début
            self.Début.Précédent = C
            self.Début = C
```



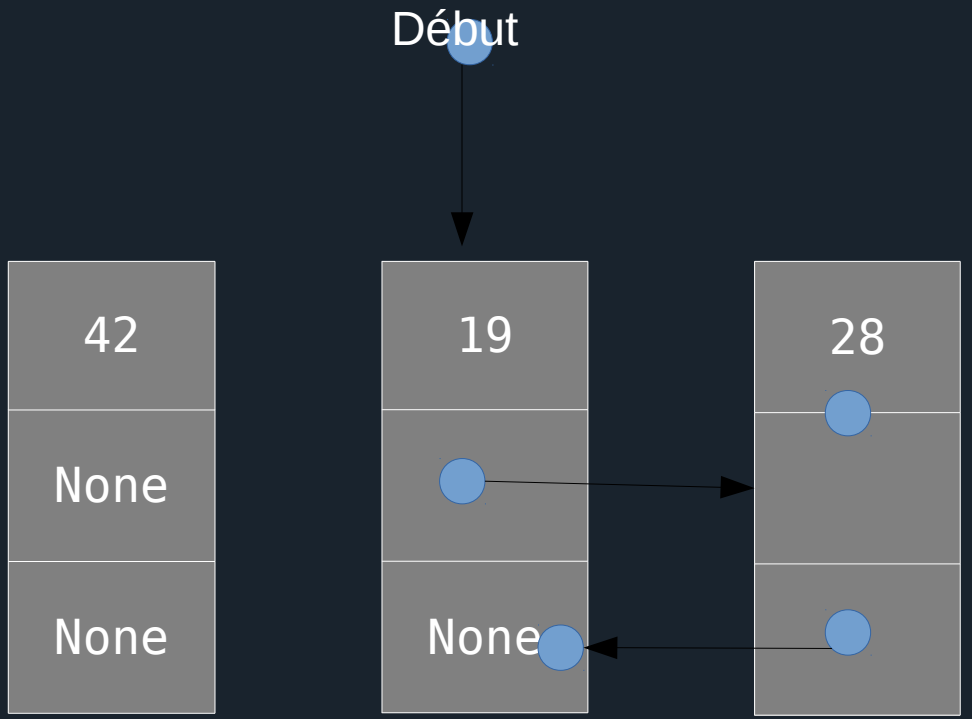
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):
        if self.Début == self.Fin == None:
            self.ajouter_dans_FIFO_vide(v)
        else:
            C = Cellule()
            C.Valeur = v
            C.Suivant = self.Début
            self.Début.Précédent = C
            self.Début = C
```



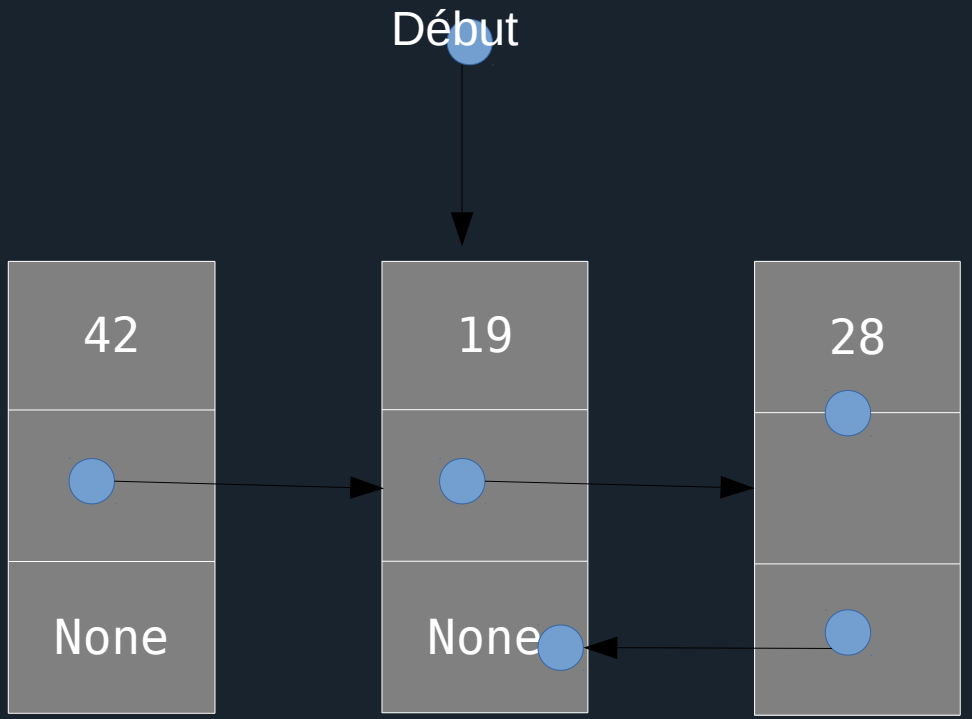
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):
        if self.Début == self.Fin == None:
            self.ajouter_dans_FIFO_vide(v)
        else:
            C = Cellule()
            C.Valeur = v
            C.Suivant = self.Début
            self.Début.Précédent = C
            self.Début = C
```



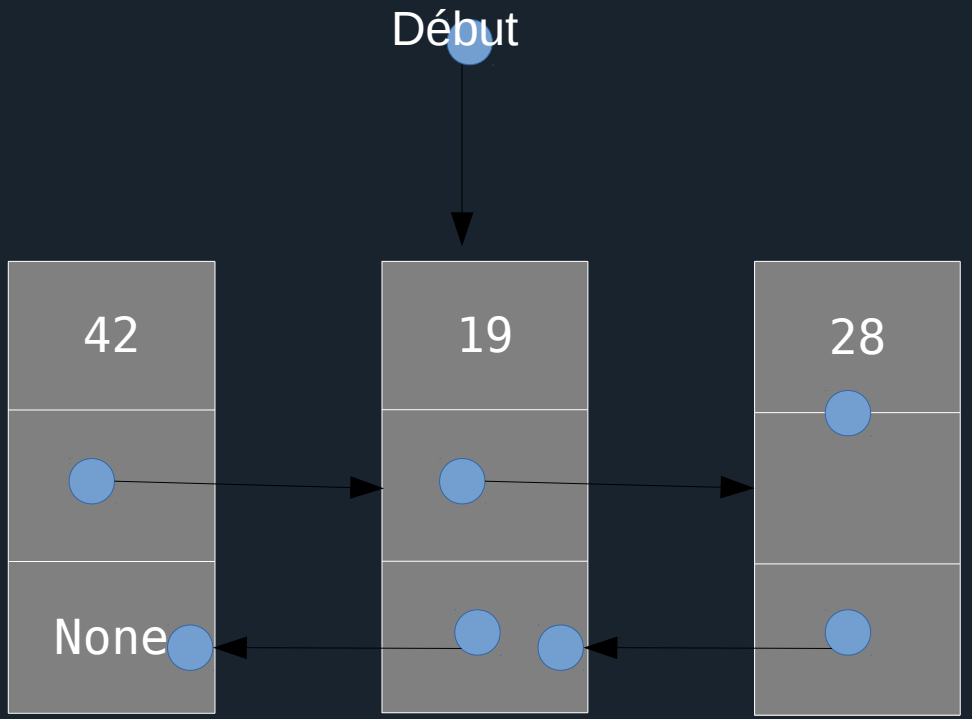
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):
        if self.Début == self.Fin == None:
            self.ajouter_dans_FIFO_vide(v)
        else:
            C = Cellule()
            C.Valeur = v
            C.Suivant = self.Début
            self.Début.Précédent = C
            self.Début = C
```



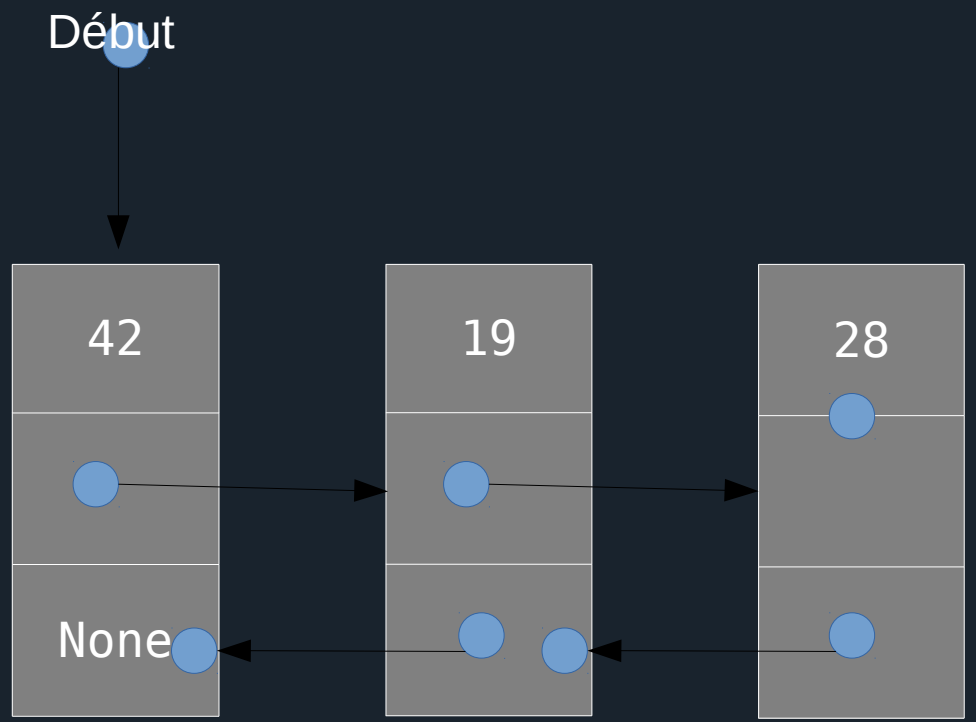
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):
        if self.Début == self.Fin == None:
            self.ajouter_dans_FIFO_vide(v)
        else:
            C = Cellule()
            C.Valeur = v
            C.Suivant = self.Début
            self.Début.Précédent = C
            self.Début = C
```



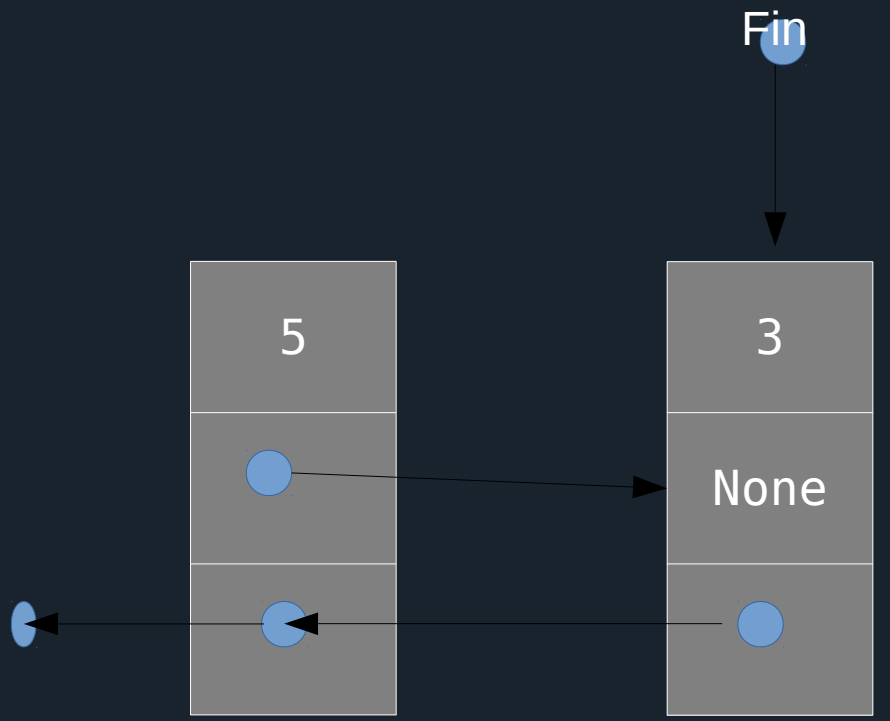
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):
        if self.Début == self.Fin == None:
            self.ajouter_dans_FIFO_vide(v)
        else:
            C = Cellule()
            C.Valeur = v
            C.Suivant = self.Début
            self.Début.Précédent = C
            self.Début = C
```



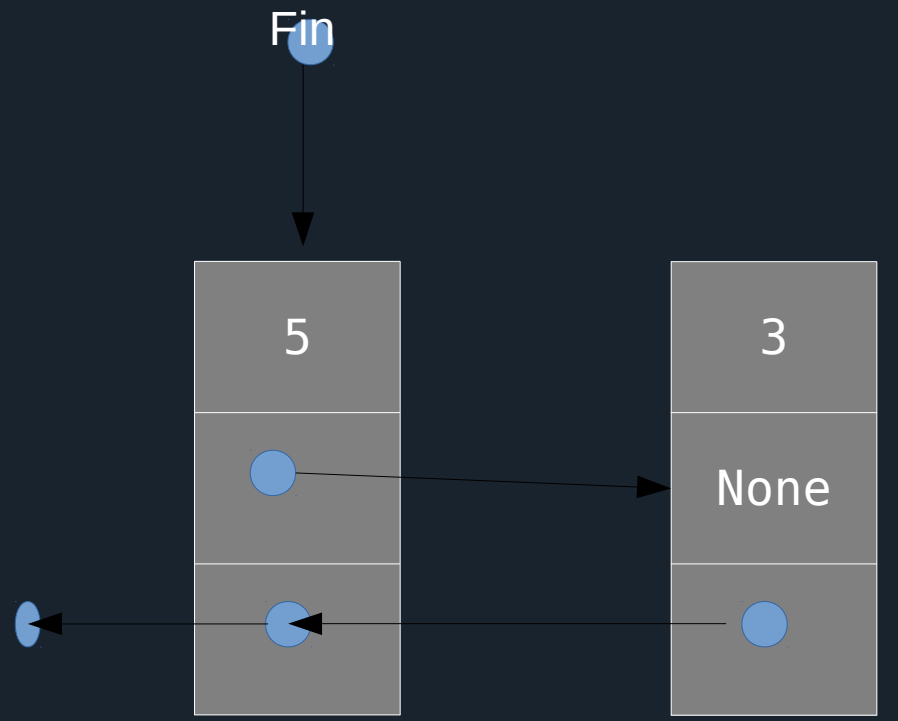
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):

    def retirer(self):
        Résultat = self.Fin.Valeur
        self.Fin = self.Fin.Précédent
        if self.Fin == None:
            self.Début = None
        else:
            self.Fin.Suivant = None
        return Résultat
```



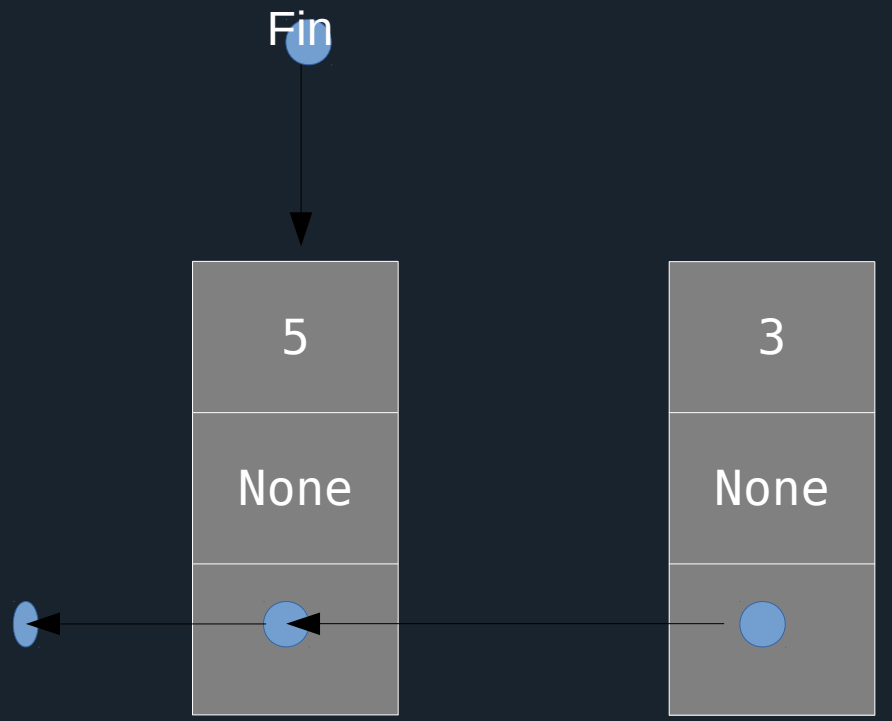
FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):

    def retirer(self):
        Résultat = self.Fin.Valeur
        self.Fin = self.Fin.Précédent
        if self.Fin == None:
            self.Début = None
        else:
            self.Fin.Suivant = None
        return Résultat
```



FIFO

```
class FIFO():
    Début = None
    Fin = None

    def ajouter_dans_FIFO_vide(self, v):

    def ajouter(self, v):

    def retirer(self):
        Résultat = self.Fin.Valeur
        self.Fin = self.Fin.Précédent
        if self.Fin == None:
            self.Début = None
        else:
            self.Fin.Suivant = None
        return Résultat
```



FIN

Au revoir !